CS 6410: Compilers

Spring 2017

HW 6 – Data Flow Analysis, Loops and SSA

Assigned: Thursday, November 21, 2019
Instructor: Tamara Bonaci
Khoury College of Computer Sciences
Northeastern University – Seattle

Submission Guidelines

- Please turn in your homework as a single .pdf file using Canvas.
- You do not have to type in your submission - hand-written and then scanned, or photographed documents are fine, as long as the total size of your document is not too big, and your document is readable.
- This assignment is meant to be worked on individually, and you should submit it by **11:59pm on Thursday, December 12, 2019** (please notice the unusual deadline - this assignment is due during the finals week).

Problem 1 (Cooper and Torczon, Problem 8.5)

Consider the following simple five-point stencil computation:

```
do 20 i = 2, n-1, 1
    t1 = A(i,j-1)
    t2 = A(i,j)
    do 10 j = 2, m-1, 1
        t3 = A(i,j+1)
        A(i,j) = 0.2     (t1 + t2 + t3 + A(i-1,j) + A(i+1,j))
                   t1 = t2
        t2 = t3
10      continue
20  continue
```

Each iteration of the loop executes two copy operations.

1. Loop unrolling can eliminate the copy operations. What unroll factor is needed to eliminate all copy operations in this loop?
2. In general, if a loop contains multiple cycles of copy operations, how can you compute the unroll factor needed to eliminate all of the copy operations?

Problem 2

Consider the control flow graph depicted in Figure 1. Compute the dominator tree for the given CFG, and then compute the dominance frontiers for nodes B2, B5, and B6.

Problem 3

Please consider the following control flow graph:
Dataflow analysis: Recall that **live-variable analysis** determines for each point p in a program which variables are live at that point. A live variable v at point p is one where there exists a path from point p to another point q where v is used without v being redefined anywhere along that path. The sets for the live variable dataflow problem are:

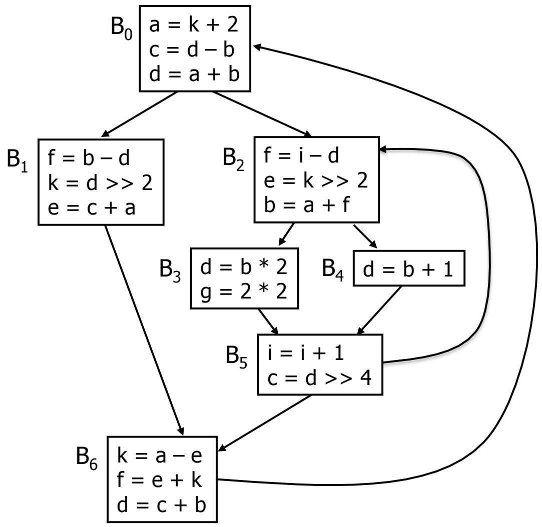

Fig. 1. Control flow graph used in Problem 2.

- **use[b]** = variables used in block *b* before any definition
- **def[b]** = variables defined in block *b*, and not later killed in *b*
- **in[b]** = variables live on entry to block *b*
- **out[b]** = variables live on exit from block *b*

The dataflow equations for live variables are given as:

$\text{in[b]} = \text{use[b]} \cup (\text{out[b]} - \text{def[b]})$
$\text{out[b]} = \cup_{s \in succ[b]}\ \text{in[s]}$

Please calculate the **use** and **def** sets for each block, then solve for the **in** and **out** sets of each block. **Hint:** remember that live-variables is a backwards dataflow problem, so the algorithm should update the sets from the end of the flowgraph towards the beginning to reduce the total amount of work needed.

Problem 4 (Cooper and Torczon, Problem 8.6)

At some point *p*, **Live(p)** is the set of names that are live at *p*. **LiveOut(b)** is just the Live set at the end of block *b*.

1. Develop an algorithm that takes as input a block *b* and its **LiveOut** set, and produces as output the **Live** set for each operation in the block.
2. Apply your algorithm to blocks *b*0 and *b*14, given below, using **LiveOut(b0)** = {t3, t9} and **LiveOut(b1)** = {t7, t8, t9}.

```
Block b0
t1 = a + b
t2 = t1 + c
t3 = t2 + d
t4 = b + a
t5 = t3 + e
t6 = t4 + f
t7 = a + b
t8 = t4 − t7
t9 = t8*t6
```

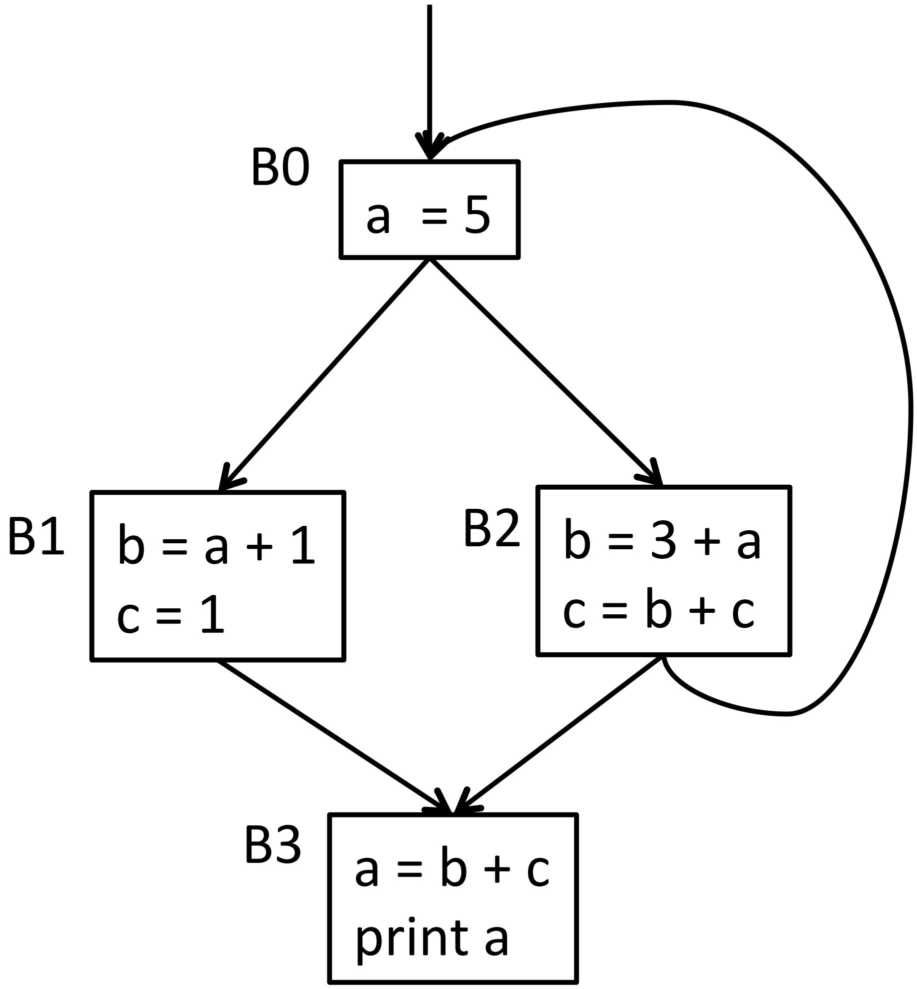

Fig. 2. Control flow graph used in Problems 3 and 5.

```
Block b1
t1 = a * b
t2 = t1 * 2
t3 = t2 * c
t4 = 7 + t3
t5 = t4 + d
t6 = t5 + 3
t7 = t4 + e
t8 = t6 + f
t9 = t1 + 6
```

Problem 5

Please consider the control flow graph from Problem 3 again. Redraw the given flowgraph in **SSA (static single-assignment) form**. You need to insert appropriate Φ-functions where they are required, and add appropriate version numbers to all variables. Do not insert Φ-functions at the beginning of a block if they clearly would not be appropriate there, but we will not penalize occasionally extra Φ-functions if they are inserted correctly.